

REMARKS

Applicants respectfully request the Examiner's reconsideration of the present application.

Claims 1, 3, and 5-22 in the present application are pending.

Claims 1, 3, 5-7, 9, and 11-22 are rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 6,711,602 ("Bhandal") in view of U.S. Patent No. 7,046,723 ("Schier").

Claims 8 and 10 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

Claims 1, 3, 5-7, 9, and 11-22 are rejected under 35 U.S.C. § 103(a) as being unpatentable over 6,711,602 ("Bhandal") in view of U.S. Patent 7,046,723 ("Schier").

Claims 1, 3, 11, 17, 21, and 22 have been amended.

Claims 23 and 24 have been added.

Support for amended claims 1, 3, 11, 17, 21, and 22 may be found at paragraphs [0003], [0005], [0028], [0037]-[0038], [0040]-[0041] in the specification, and Figures 2 and 5-6 in the drawings. No new matter has been added.

Support for new claims 23 and 24 may be found at paragraphs [0049]-[0050], and [0052], and Figures 5, 7-8, and 9-10 in the drawings. No new matter has been added.

It is submitted that Bhandal and Schier do not render claims 1, 3, 5-7, 9, and 11-24 unpatentable under 35 U.S.C. §103(a).

Bhandal includes a disclosure of a pair of parallel 16.times.16 multipliers each with two 32-bit inputs and one 32-bit output. There are options to allow input halfword and byte selection for four independent 8x8 or two independent 16x16 multiplications, real and imaginary parts of complex multiplication, pairs of partial sums for 32x32 multiplication, and partial sums for

16x32 multiplication. There are options to allow internal hardwired routing of each multiplier unit results to achieve partial-sum shifting as required to support above options. There is a redundant digit arithmetic adder before final outputs to support additions for partial sum accumulation, complex multiplication vector accumulation and general accumulation for FIRs/IIRs--giving MAC unit functionality. There are options controlled using bit fields in a control register passed to the multiplier unit as an operand. There are also options to generate all of the products needed for complex multiplication (see Bhandal Abstract).

Schier includes a disclosure of a digital and a multiplication method are described, which lead to an efficient architecture for a hardware implementation of digital FIR and IIR filters into FPGAs. The multiplications of input sample data and delayed sample data with filter coefficients are performed by addressing look-up tables in which corresponding multiplication results are prestored. The size of the look-up tables is reduced by storing only those multiplication results which cannot be obtained by a shifting operation performed on the other pre-stored multiplication results, the input sample data, or the delayed sample data. Thereby, the size of the look-up tables can be compressed significantly such that an implementation of large digital filters into FPGAs is possible (see Schier Abstract).

In the Office Action mailed 6/10/2009, the Office states in part the following.

Bhandal et al. fail to disclose the multiplier is on a field programmable gate array and that the second product is retrieved from a memory. However, Schier et al. disclose in Figures 1-4 the multiplier is on a field programmable gate array (e.g. abstract) and the second product is retrieved from a memory (e.g. any intermediate product from the LUT in Figures 1-4 as blx).

...

The examiner respectfully submits that the detail as how the primary reference by Bhandal anticipates the limitations in claim 1 is clearly seen in the above rejection. In addition, by definition the subset of the set can also be the set, thus Bhandal does not need to show or disclose the multiplication is done for only the subset of input bits of operands. Even if the subset is a set less than the set, Bhandal clearly discloses that in the Figures and the abstract wherein the two 16-bit multipliers can operate as 4

independent 8x8 multiplications for 32-bit operands. Thus, at anytime the pair of multipliers only operates only 16 bits out of 32 bit operand (e.g. two 8x8 multiplication operations at a time). Thus, the total 16 bits operation is a subset of the 32 input bit operands.

(6/10/2009 Office Action, pp. 3, and 12-13) (Emphasis Added).

It is submitted that Bhandal and Schier do not teach or suggest a method for performing multiplication of a first number with a second number on a field programmable gate array that includes generating a product by multiplying a first plurality of bits from the first number and a first plurality of bits from the second number using a digital signal processor (DSP) only capable of supporting multiplication on a number of bits that are fewer in number than that forming the first and second numbers, retrieving a stored value designated as a product of a second plurality of bits from the first number and a second plurality of bits from the second number from a memory where the second plurality of bits from the first number is fewer than the bits of the first number and the second plurality of bits from the second number is fewer than the bits of the second number, scaling the product with respect to a position of the first plurality of bits from the first number and a position of the first plurality of bits from the second number and scaling the stored value with respect to a position of the second plurality of bits from the first number and a position of the second plurality of bits from the second number, and summing a scaled product and a scaled stored value to generate a value representing a product of the first number and the second number, wherein the first number and the second number each have a number of bits equal to or greater than a total of the first and second plurality of bits.

Firstly, Bhandal and Schier do not teach or suggest generating a product by multiplying a first plurality of bits from the first number and a first plurality of bits from the second number using a digital signal processor (DSP) only capable of supporting multiplication on a number of bits that are fewer number than that forming the first and second numbers.

On the contrary, Bhandal discloses a digital signal processor 44 (shown in Figures 1 and 2) that includes a plurality of M-unit groups 84 each having a M Galois multiply unit 164 and a M multiply unit 171 (shown in Figure 5). Figure 11A-I illustrate various multiply operations performed by the multiply units including two 16x16 and four 8x8 multiplies (see Bhandal column 5, lines 11-15, column 6, lines 15-27, column 8, lines 3-10, column 8, line 40 through column 9, line 4 and Figures A-I). In all of the multiply operations described and illustrated in Bhandal, the DSP 44 in Bhandal is capable of supporting multiplication on all the bits forming the first and second numbers when using a plurality of multipliers that multiply a smaller set of bits forming the first and second numbers. Furthermore, Schier only discloses a digital filter and method for performing a multiplication based on a look-up table.

Secondly, Bhandal and Schier do not teach or suggest retrieving a stored value designated as a product of a second plurality of bits from the first number and a second plurality of bits from the second number from a memory where the second plurality of bits from the first number is fewer than the bits of the first number and the second plurality of bits from the second number is fewer than the bits of the second number.

On the contrary, the Office acknowledges that Bhandal does not disclose retrieving a stored value designated as a product from a memory (see 6/10/2009 Office Action, p. 3). Furthermore, Schier discloses a LUT based multiplier where the entire result of multiplication of sample data with filter coefficients are prestored. Input sample data $x(n)$ is not split up into their bit positions, but are completely supplied to the LUT based multiplier 2 in order to address a look-up table corresponding to the respective filter coefficient and pre-storing result of multiplications (see Schier column 5, lines 29-35, and Figure 1). Schier does not prestore only a portion of a result. Weighted sample data generated from multiplying sample data with filter coefficients are outputted from the LUT based multiplier 2 (see Schier column 7, line 65 through column 8, line 29 and Figures 1 and 4).

Thirdly Bhandal and Schier also do not teach or suggest summing a scaled product and a scaled stored value to generate a value representing a product of the first number and the second number.

On the contrary, Bhandal only discloses summing the outputs of multipliers, not a scaled product and a scaled stored value to generate a value representing a product (see Bhandal column 2, lines 53-60, Figures 11B, 11D, 11E, 11 F, and 11H). Furthermore, Schier only discloses summing weighted sample data, not a scaled product and a scaled stored value to generate a value representing a product. The weighted sample data are not summed by adder 3 to generate a product of a first number and a second number, but to obtain filter output data $y(n)$ (see Schier column 5, lines 46-52).

Moreover, Applicants submit that there is no basis for combining Bhandal and Schier. The Office acknowledges that Bhandal does not disclose “retrieving a stored value designated as a product of a second plurality of bits from the first number and a second plurality of bits from the second number from a memory” as required by the present invention.

In the Office Action mailed 6/10/2009, the Office states in part the following.

Bhandal et al. fail to disclose the multiplier is on a field programmable gate array and that the second product is retrieved from a memory. However, Schier et al. disclose in Figures 1-4 the multiplier is on a field programmable gate array (e.g. abstract) and the second product is retrieved from a memory (e.g. any intermediate product from the LUT in Figures 1-4 as blk).

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention is made to add the multiplier is (sic) on a field programmable gate array and the second product is retrieved from a memory as seen in Schier et al.’s invention into Bhandal et al.’s invention because it would enable to improve system performance (e.g. col. 3 lines 9-11 and col. 4 lines 9-12)

(6/10/2009 Office Action, pp. 3-4) (Emphasis Added).

Applicants submit that the Office cannot properly combine “any intermediate product from the LUT in Figures 1-4” of Schier with Bhandal to be used as “the second product” as the

Office suggests on page 3 of the Office Action mailed 6/10/2009. In Schier, the values $b0x(n)$ to $b4x(n-4)$ are not “intermediate products”. The values are final products that do not require further processing to generate a multiplication product. Adder 3 sums the multiplication products $b0x(n)$ to $b4x(n-4)$ to generate filter output data $y(n)$, not to generate a multiplication result.

Applicants also disagree with the Office’s motivation for combining Bhandal with Schier when the Office states that “it would have been obvious to a person having ordinary skill in the art at the time of the invention is made to add ... the second product is (sic) retrieved from a memory as seen in Schier et al.’s invention into Bhandal et al.’s invention because it would enable to improve system performance.” Applicants submit that the example in Bhandal discloses that the number of bits of the numbers being multiplied, 32, match the configuration of the number of bits multiplied by multipliers 800 and 801, 32. Thus, there is no need or motivation to add a further product from memory. Furthermore, because there is a match, there would be no improvement in system performance. In fact, by requiring a stored product to be added, the multiplier in Bhandal would have its performance worsen not improve.

In the Advisory Action mailed 10/21/2009, the Office states in part the following.

The applicant further argues in pages 10-12 for claims that there is no motivation as “improving system performance” for combination since there is no need to add a further product from memory and also requiring the stored product to be added, the multiplier in Bhandal would have its performance worsen not improved as stated by the examiner.

The examiner respectfully submits that it is conventionally known the product LUT is improvement over the direct multiplication (certain degree). By having one of the multiplications replace with direct lookup table to obtain the product would obviously improve over direct multiplication operation.

(10/21/2009 Advisory Action, p. 2) .

Applicants respectfully requests that the Office provide documentary evidence in the next Office Action as is in accordance with MPEP §2144.03(C) (“If Applicant Challenges a

Serial No. 10/829,559

13

ALT.P030 (A01252)

Factual Assertion as Not Properly Officially Noticed or not Properly Based Upon Common Knowledge, the Examiner Must Support the Finding with Adequate Evidence"). Specifically, Applicants respectfully requests that the Office provide documentary evidence and specifically point out teachings or suggestions that it is conventionally known that multiplication based on LUT is an improvement over direct multiplication.

If the Office is relying on personal knowledge to support this finding, an affidavit or declaration setting forth specific factual statements and explanation to support this finding is respectfully requested (MPEP §2144.03 (C)).

In contrast, claim 1 states

A method for performing multiplication of a first number with a second number on a field programmable gate array, comprising:

generating a product by multiplying a first plurality of bits from the first number and a first plurality of bits from the second number using a digital signal processor (DSP) only capable of supporting multiplication on a number of bits that are fewer number then those forming the first and second numbers;

retrieving a stored value designated as a product of a second plurality of bits from the first number and a second plurality of bits from the second number from a memory where the second plurality of bits from the first number is fewer than the bits of the first number and the second plurality of bits from the second number is fewer than the bits of the second number;

scaling the product with respect to a position of the first plurality of bits from the first number and a position of the first plurality of bits from the second number and scaling the stored value with respect to a position of the second plurality of bits from the first number and a position of the second plurality of bits from the second number; and

summing a scaled product and a scaled stored value to generate a value representing a product of the first number and the second number, wherein the first number and the second number each have a number of bits equal to or greater than a total of the first and second plurality of bits.

(Claim 1, as Amended) (Emphasis Added).

Claims 11, 17, 21, and 22 include similar limitations.

Given that claims 3, 5-10, and 23-24 depend from claim 1, claims 12-16 depend from claim 11, and claims 18-20 depend from claim 17, it is likewise submitted that claims 3, 5-10, 12-16, 18-20, and 23-24 are also patentable under 35 U.S.C. §103(a) over Bhandal and Schier.

Applicants submit that Bhandal and Schier also do not teach or suggest scaling a product with respect to a position of the first plurality of bits from the first number and a position of the first plurality of bits from the second number wherein scaling the product comprises routing the product directly to an adder at inputs of appropriate significance.

On the contrary, the Office acknowledges that Bhandal scales a product "by shifter 810 in Figure 8" (6/10/2009 Office Action, p. 3). Bhandal illustrates routing an output of multiplier 800 to a shifter 810 before routing the output to adder/converter 820 (see Bhandal column 7, lines 43-50 and Figure 8). The product from multiplier 800 is not scaled by being routed directly to adder/converter 820 at inputs of appropriate significance.

Schier only discloses a method for performing a multiplication based on a look-up table. Schier does not teach or suggest scaling a product with respect to a position of the first plurality of bits from the first number and a position of the first plurality of bits from the second number wherein scaling the product comprises routing the product directly to an adder at inputs of appropriate significance.

In contrast, claim 23 states

The method of Claim 1, wherein scaling the product
comprises routing the product directly to an adder at inputs of
appropriate significance.

(Claim 23, Emphasis Added).

Claims 11 and 21 include similar limitations.

Applicants submit that Bhandal and Schier also do not teach or suggest scaling a stored value with respect to a position of the second plurality of bits from the first number and a

position of the second plurality of bits from the second number wherein scaling the stored value comprises routing the stored value directly to an adder at inputs of appropriate significance.

On the contrary, Schier discloses a LUT 2 that routes output data stored from a reduced table 210/215 directly to a bit shifting units 220/225 where a bit shifting operation is performed (see Schier column 8, lines 18-23 and Figure 4). The out data is transmitted to the bit shifting units 220/225 before being output to adder 3.

Furthermore, the Office acknowledges that Bhandal fails to disclose a product retrieved from memory (6/10/2009 Office Action, p. 3). Bhandal does not teach or suggest scaling a stored value with respect to a position of the second plurality of bits from the first number and a position of the second plurality of bits from the second number wherein scaling the stored value comprises routing the stored value directly to an adder at inputs of appropriate significance. In contrast, claim 24 states

The method of Claim 1, wherein scaling the stored value comprises routing the stored value directly to an adder at inputs of appropriate significance.

(Claim 24, Emphasis Added).

Claims 11 and 21 include similar limitations.

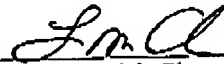
In view of the arguments set forth herein, it is respectfully submitted that the applicable rejections and have been overcome. Accordingly, it is respectfully submitted that claims 1, 3, and 5-24 should be found to be in condition for allowance.

The Examiner is invited to telephone Applicants' attorney (217-377-2500) to facilitate prosecution of this application.

If any additional fee is required, please charge Deposit Account No. 50-1624.

Respectfully submitted,

Dated: January 28, 2010


Lawrence M. Cho
Attorney for Applicants
Registration No. 39,942

P.O. Box 2144
Champaign, IL 61825
(217) 377-2500